Computer Vision: Face Detection Algorithm

i) Motivation:

A computer vision algorithm to detect and identify individuals, using a database of known face images (specifically, students in the module) has been developed. The dataset consisted of 48 students with individual IDs and a group of students in the lecture, some of whom were not assigned individual IDs and were 'unknown' to the model. The dataset consisted of both individual students with ID's, and a group of unidentified students; with their images in JPEG format and videos in mp4 format.

ii) Data Pre-Processing

❖ Face-Detection

First, individuals' images and videos with their corresponding IDs' were subset in their respective folders. From each video of each individual ID, about 30-40 frames were extracted and saved in the respective individual folders. Subsequently, all images were mean and variance normalised to compensate for different levels of brightness in the images. Following this step, from the individual frames, faces were detected and cropped using the Viola Jones face detection algorithm.

The Viola-Jones algorithm uses Haar-like features, that is, a scalar product between the image and some Haar-like templates. That is, let I and P denote an image and a pattern, both of the same size N ×N; the feature associated with pattern P of image I is defined by (1):

$$\sum_{1<i,j<N} \boldsymbol{I(i,j) * P(i,j)\ is\ white} - \sum_{1<i,j<N} \boldsymbol{I(i,j) * P(i,j)\ is\ black}$$

Viola Jones algorithm uses five Haar-like pattern features to detect faces, since a face is by and large regular in nature; this choice seemed justified. Further, instead of summing up all the pixels inside a rectangular window, this technique mirrors the use of cumulative distribution functions; making it computationally more efficient. The cropped faces were stored in the folders matching with their corresponding IDs (1).

Additionally, using the students in the unidentified group images, using still frames (JPEG format) about five frames per student ID were extracted using Viola-Jones algorithm; and manually mapped to the respective ID folders. These group photos also consisted of unknown students (without IDs) which weren't included in our dataset. The consideration of faces from group images was particularly vital cause they comprised of different conditions as compared to the individual pictures, henceforth changing the distribution of the data. This added heterogeneity to the dataset; particularly since models had to be tested in the group image environment.

❖ Data Augmentation

Going forward, the image data augmentation was performed in order to increase the ability of data to generalise under different conditions before model building. This included Gaussian function to blur an image, performing a random degree of rotation between (-90 & 90) on the image and a function to flip the image (Fig-1).

Fig1: (a)Original (b)Gaussian Blur (c)Flip (d) Rotation

iii) Feature extraction:

Next, we perform feature extraction on the aforementioned image dataset. This is a dimensionality reduction technique that represents key information of the image as a compact vector and removes the extraneous information. For feature extraction two techniques are predominantly used 1) Global feature extraction which uses the image as a whole to extract features as a single vector, or 2) local feature descriptors, which are calculated at different points in the image and typically represent texture in an image patch. Due to underlying differences in the way the two descriptors are calculated, they provide different information of the images and we explore both features on our dataset (2).

❖ Local Feature extraction

i. Histogram of Oriented gradients (HoG)

To explore local feature extraction, we used Histogram of Oriented gradients (HoG) technique. Each image was resized to a homogenous size of 128*64, following which gradients along x-axis and y-axis of the image were calculated. At every pixel, the gradient has a magnitude and a direction. The direction points to the direction of change in intensity, and the magnitude shows how big the difference is. The magnitude of the gradient changes with the sharp change in intensity in every pixel. For colour images, the gradients of the three channels are evaluated and the magnitude of gradient at a pixel is the maximum of the magnitude of gradients of the three channels, and the angle is the angle corresponding to the maximum gradient.

For calculating HOG, the image is divided into 8*8 cells. The histogram is essentially a vector of 9 bins; corresponding to angles 0, 20, 40, 60 … 160. The next step is to create a histogram of gradients in these 8×8 cells. For each cell, the histogram bin is selected based on the direction of the gradient vector, and the value that goes into the bin is selected based on the magnitude. The contributions of all the pixels in the 8×8 cells are added up to create the 9-bin histogram. Fig2 below gives an example of direction and magnitude of the gradient vector for a cell, and displays the corresponding division into histogram bins.

Next in order to remove sensitivity of different patches of the image to overall lighting, the gradients are normalised by taking 16×16 blocks. A 16×16 block has 4 histograms which can be concatenated to form a 36 x 1 element vector, and then window is then moved by 8 pixels again and the process is iteratively repeated. To calculate the final feature vector for the entire image patch, the 36×1 vector is concatenated into one giant vector.

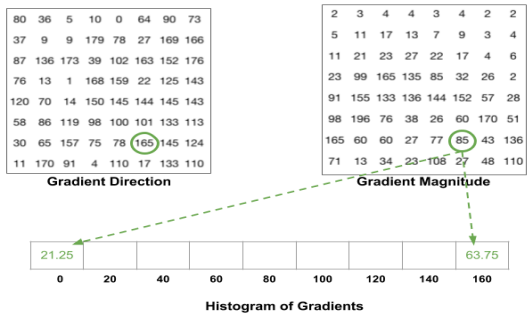


Fig2: Magnitude and direction of gradients divide into Histogram Bins

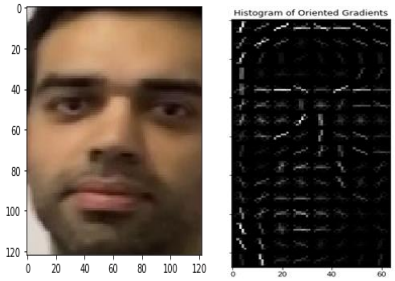


Fig3) Original and Hog Image

❖ Global feature extractors

For the purpose of our study, we examine three global feature vectors and concatenate them to form a single feature vector

i) Colour:

First, we created a histogram for HSV channels by discretizing the HSV into 8 bins each; and histogram was made by counting the number of image pixels in each bin. The pixel values in each bin were first normalized and then we quantized the coordinates into each bin. The color histogram of an image is

relatively invariant with translation and rotation about the viewing axis, and varies only slowly with the angle of view(3).

ii) Shape

Hu-moments were used to study the shape of an image. An image moment is a certain particular weighted average of the image pixels' intensities, or a function of such moments, usually chosen to give an interpretation of the shape ie) area (or total intensity), its centroid and orientation of the image. In our study we use Hu-Moments, which are the two-dimensional moment invariants with respect to the fixed pair of axes, which helps to achieve orientation independence without ambiguity, by using either absolute or relative orthogonal moment invariants. The Hu moments are independent of the pattern position in the visual field and also independent of the pattern size(4).

iii)Texture

For texture description we used Harlick texture descriptor which is a quantification of the spatial variation of grey tone values. This uses gray level co-occurrence matrices (GLCM) which constitutes the joint probability distributions of pairs of pixels. GLCM show how often each gray level occurs at a pixel located at a fixed geometric position relative to each other pixel, as a function of the gray level (Srinivasan and Shobha 2008). An essential component is the definition of eight nearest-neighbour resolution cells that define different matrices for different angles (0°,45°,90°,135°) and distances between the horizontal neighbouring pixels (Fig4)(5).

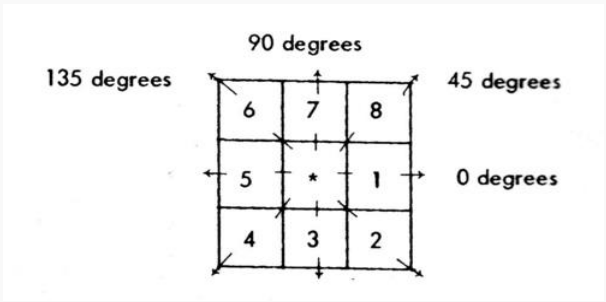


Fig4: 3x3 window definition and spatial relationship for calculating Haralick texture measures. Pixel 1 and 5 are 0° (horizontal) nearest neighbours to the center pixel *; pixel 2 and 6 are 135° nearest neighbours; pixels 3 and 7 are 90° nearest neighbours, pixel 4 and 8 are 45° nearest neighbours to the centre pixel * (Haralick et al. 1973)

iv) Model Construction: *Explored 4 classification Algorithms on Local & Global features:*

The dataset consisted of 4616 cropped face images with 48 recognised students with IDs, with a different number of images in each student class. For both the local and global extracted features, 70% data was used as training data and 30% was our hold out test data, based on a stratified split of the data (given the class imbalance). On the training data, 3-fold stratified-cross validation was performed for the purpose of hyper parameter tuning for model selection. Further, given the class imbalance both F1 score and accuracy were used to determine the chosen models. **We investigated four machine learning classification algorithms on both global and local image features in order to examine the given image classification problem.**

❖ Logistic Regression:

This is a classification algorithm used when dependent variable is qualitative and the goal of the analysis is a classification task. The logistic function, also called the sigmoid is an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1. When the dependent variable consists of more than two classes, then multinomial logistic regression is used It uses the softmax function instead of the sigmoid function which compresses all values to the range [0,1] and the sum of the elements is 1. As the name suggests, in softmax regression (SMR), we replace the sigmoid logistic function by the so-called softmax function ϕ:

$$P(y = j \mid z^{(i)}) = \phi_{softmax}(z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{j=0}^{k} e^{z_k^{(i)}}},$$

where we define the net input z as:

$$z = w_0 x_0 + w_1 x_1 + \ldots + w_m x_m = \sum_{l=0}^{m} w_l x_l = \mathbf{w}^T \mathbf{x}.$$

(w is the weight vector, x is the feature vector of 1 training sample, and w0 is the bias unit.) Now, this softmax function computes the probability that this training sample x(i) belongs to class j given the weight and net input z(i). So, we compute the probability p (y = j | x(i), wj) for each class label in j = 1, ..., k.

PROS • Fast and simple to implement probabilistic model, with high interpretability, which explains its low variance and high bias. (6) •Can be made more computationally efficient by using optimization algorithms such as Stochastic Gradient Descent (7). • Quite robust to imbalanced data, especially if the training sample is representative of the real distribution of events. **CONS** • Logistic Regression, while robust, tends to be less accurate than ensemble methods, especially to map complex dynamics. Although interactions can increase the sophistication of the models, they are limited in terms of complexity as seen from their simplistic linear decision boundaries. • The performance of a logistic regression, as a discriminative model, depends mainly on the quality and representativeness of the training data.

❖ Support-Vector-Machines:

The Support Vector machine algorithm finds a N-dimensional hyperplane in order to classify the dataset with the use of support vectors. The SVM maximises the margin of the hyperplane, i.e.) maximises the distance between hyperplane and the datapoints in the classes. Additionally, SVM allows us to convert not separable problem to separable problem by using the kernel function encompassing the kernel trick, which allows us to take a low dimensional input space and transforms it to a higher dimensional space.

For SVM typically radial basis function kernel and Polynomial kernel functions are evaluated. The hyper parameters are C; which is a regularization parameter that controls the trade-off between the achieving a low training error and a low testing error, that is the ability to generalize your classifier to unseen data (bias-variance trade off). In case of RBF kernel, gamma hyper-parameter which is the inverse of the standard deviation of the RBF kernel (Gaussian function) needs to be optimized. For, polynomial kernel, the degree of the polynomial is the hyper-parameter.

Pros: • It is a quadratic programming problem that is computationally efficient (8). • It is effective in cases where number of dimensions is greater than the number of samples and works even well with small datasets. **CONS:** • SVM shows poor performance on noisy data with overlapping classes. • The classifying hyper plane offers no probabilistic explanation.

❖ Random Forests:

Builds an ensemble of weak learning trees with the trees grown sequentially. Each tree is grown using information from a bagged subset of previously grown trees. Further in our analysis we examined boosting, which focuses on incrementally reducing bias, by reweighting training examples for the next ensemble model, based on a measure of error for the current ensemble of models. The hyper parameters considered were number of trees, minimum samples per split, minimum leaf size and maximum depth, which were carefully chosen taking care of the bias-viariance trade-off. Further bootstrapping was performed to prevent overfitting.

PROS: • It ranks features according to their importance, which some other models can't do. **CONS** • In case of class imbalance, boosting algorithms may suffer from bias towards majority class, since their

learning process is guided by their loss function and therefore correct classification, implicitly giving more weight to correct classification of majority class. • It searches a less restricted space of models, allowing it to capture nonlinear patterns in the data, but making it less stable and prone to overfitting, especially with weak classifiers that are too complex.

❖ K nearest neighbours:

The k-nearest neighbours algorithm is a supervised classification algorithm that takes labelled points and uses them for classification of other points. To classify a new point, it looks at the labelled points closest to that new point which are its nearest neighbours. So whichever label, the most of the neighbours have is the label for the new point.

The hyper parameter considered was k, ie) the number of neighbours chosen for the aforementioned classification task. The choice of K encompasses a delicate balance. A value of k too low and lead to overfitting and too large a value might lead to high bias.

Pros: • K-NN is a non-parametric algorithm which means there are assumptions to be met to implement K-NN. •Given it's an instance-based learning; k-NN is a memory-based approach. The classifier immediately adapts as we collect new training data. •It allows the algorithm to respond quickly to changes in the input during real-time use. **Cons:** •As dataset grows, it becomes more and more computationally inefficient. • It suffers from the Curse of Dimensionality.

v) Results

We performed 3-fold cross validation for both hog and global features for hyper-parameter tuning. The following parameters in grid-search were considered.

- *Hyper-parameters considered for SVM= {'C': [1, 10, 100, 1000], 'kernel': ['linear']}, {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']},*
- *Hyper-parameters considered for Random Forests: { 'bootstrap': [True], 'max depth': [80, 90, 100, 110],*
- *'min leaf size': [3, 4, 5], 'min samples per split': [8, 10, 12], 'no. of trees ': [100, 200, 300, 1000]}*
- *Hyper-parameters for KNN:{ leaf size = [1, 15, 20, 30], no. of neighbours = [3, 5, 7, 10]}*
- *Hyper-parameters for Logistic Regression :{'C': [1, 4, 10, 100]}*

Hyper-Parameter Selection								
	SVM		Random Forests				Logistic-Regression	KNN
Feature Type/Model	Kernel	C	No. of Tress	Max depth	Min leaf size	Min samples per split	C	k
Local Features	Linear	1	1000	100	3	8	10	3
Global Features	Linear	2	1000	110	2	6	4	3

Fig 5: Hyperparameter table for global and local feature extractors, which were chosen by 3-fold cross validation, to maximise validation dataset accuracy

Performance on Hold-Out Test Data for Individual Images								
	SVM		Random Forests		Logistic-Regression		KNN	
Feature Type/Model	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1
Local Features	0.97	0.97	0.89	0.78	0.96	0.96	0.92	0.88
Global Features	0.95	0.93	0.94	0.9	0.95	0.93	0.91	0.86

Fig 6: Performance in terms of F1 score and accuracy for 30% hold out test data for global & local features

As can be seen in the table above, except for Random Forests model; local feature extraction outperforms global feature extraction, in terms of both F1 score and accuracy on hold-out test data.

❖ Why local feature extraction out-performs global feature extraction on face recognition data?

Global feature methods are not well suited for real world applications as they encounter serious problems when dealing with local distortions that face images commonly undergo in unconstrained settings (9). On the contrary, local features are largely robust to local distortions like illumination, pose and occlusions, allowing local matching techniques to reach high recognition accuracy on face images acquired in unconstrained settings. Further, they're likely to recognise the image despite significant clutter and occlusion. They also do not require a segmentation of the object from the background, unlike many global texture or shape features. Global features suffer from sensitivity to noise, illumination variation, scaling, and henceforth fail to identify the important features of the image(10).

❖ Experimental Results

- **Support Vector Machine with linear kernel; and gamma value 1, outperformed all other classification methods, closely followed by logistic regression, for both local and global features, in terms of test performance on the hold out data.**
- **Hog SVM outperformed all other methods in terms of both F1 score and accuracy, with both at 97% on hold-out test data.**

vi) Convolutional Neural Networks:

Next, we examine Convolutional Nueral Netwroks on our dataset. CNN is a deep learning model which consists of a feed-forward artificial neural network or multi-layer perceptrons(MLPs).MLP is a multilayer feedforward deep neural network that consists of an interconnection of perceptrons with an input layer, an output layer (that makes a decision or prediction about the input); and in between there are the hidden layers. In case of CNN first the image is passed through a convolutional layer. This convolutional layer is connected to small local regions in the input image. Each convolutional layer computes a dot product between their weights and a small receptive field to which they are connected to, in the input image. The filter then produces convolution, i.e. moves along the input image and all these multiplications are summed up. Since the filter has only a local part of the image, it performs the filter operation throughout the image, post which a matrix smaller than the input is obtained.

The CNN consist of several convolutional networks, and layers; output of the first layer becomes input of the next, which continues with every future convolutional layer. After each convolution the CNN comprises of an activation function which is composed in the non-linear layer. The non-linear layer is followed by the pooling layer, and it performs a down sampling operation on the image, which reduces the volume of the image by removing features which have been already extracted by previous layers of the image. The completion of series of convolutional, nonlinear and pooling layers, is followed by a fully connected layer which results in an N dimensional vector, where N is the amount of classes from which the model selects the desired class.

❖ Architecture of CNN constructed for our Face Recognition Problem

- We applied CNN to our dataset of 4616 images where each image is a coloured image of a different size, representing is 3-dimensional coloured array. The image matrix was converted to an array, rescaled between 0 and 1 and reshaped, so that was of size 64 x 64 x 3, and fed as an input to the network.
- We proceed with 70% training data and 30% hold out test data, as in the previous models. The training data had a size of and test data has a size of 3231*64*64*3 and test data a size of 1385*64*64*3. Next, we split the training data into two parts, 80% of it was used for training and another 20% for validation.
- We used a batch size of 64 and trained the network for 20 epochs using the Rectified Linear Units (ReLUs) activation function which helped the network learn non-linear decision boundaries that could separate the 48 classes which are not linearly separable.

- The last layer was a Dense layer that had a softmax activation function with 48 units, needed for this multi-class classification problem.
- After the model was created, we compiled it using the Adam optimizer and used the categorical cross entropy loss function. Additionally, we complemented the model with a dropout layer to overcome the problem of overfitting which randomly turns off a fraction of neurons during the training process.

Fig 7: Architecture of the Model

❖ Results

- When the aforementioned CNN was trained on our dataset, the validation loss was 11% and validation accuracy was 97.6%
- Validation loss and accuracy closely mirrored the training loss and accuracy (Fig 8), thus there was no problem of overfitting.
- Even though the validation loss and accuracy line were not linear, but it showed that our model was not overfitting with validation loss & accuracy not further from training loss & accuracy.
- **Test loss on 30% hold-out data was 15% and test accuracy was 97%.**

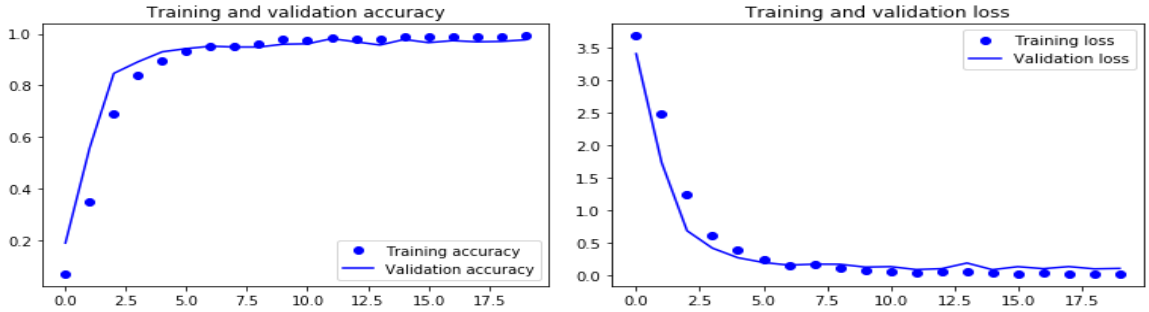


Fig 8: Validation loss and accuracy closely mirroring training loss and accuracy

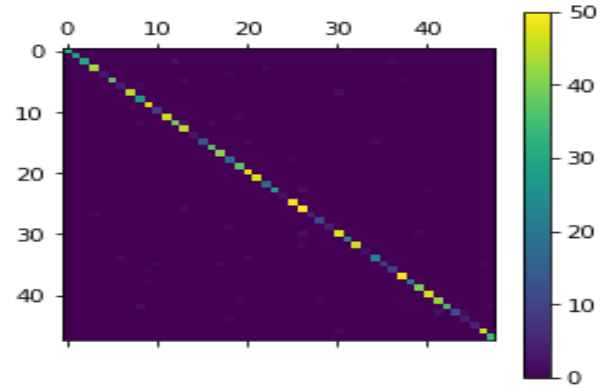


Fig 9: Confusion Matrix for CNN on test data

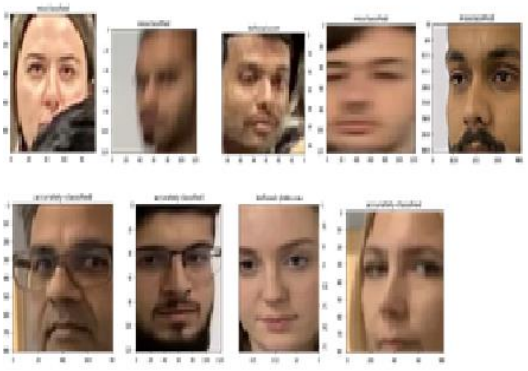


Fig 10: Top panel: misclassified images by CNN (test data)
Bottom Panel: Correctly classified images by CNN (test data)

vii) Comparison of two best performing Models: Hog SvM and CNN

❖ Performance on individual Images

- When we compared the performance of Hog SVM and CNN on 30% hold-out test data on individual face images both CNN and HoG SVM displayed the test accuracy score of 97%; meanwhile Hog-SVM displayed F1 score of 97%, higher than 95% displayed by CNN.
- In case of CNN while precision was above 80% for all classes, recall displayed high dispersion. Class 5 and class 40 showed poor recall below 75%.
- In case of SVM, all classes showed precision over 90%, however class 5 and class 40 showed very poor recall below 70%.

While for most classes both our models succeeded in predicting positive identifications as actually correct (high precision); for students 5 and 40, both models gave a low proportion of actual positives identified correctly (low recall). Further recall displayed significant dispersion for both models.

❖ Performance on group Images

Next, we see the performance of SVM and CNN on unseen group images. **If a student ID prediction was less than 15% by the models, we gave the image a value of 'unknown' below this threshold.** We ran HoG-SVM and CNN on three group images and got the following results

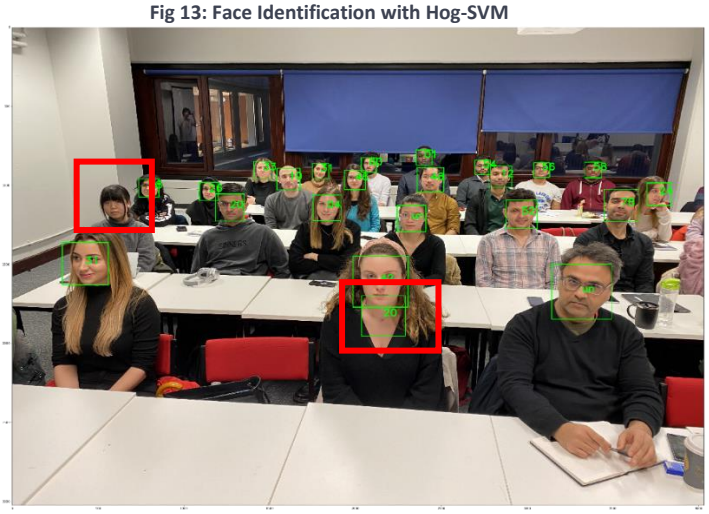
Fig 11: Face Identification with Hog-SVM

- Only one individual was wrongly classified (ID 11 identified as 14),
- At a threshold of 0.15, many known students were given 'unknown ID' and unknown students were identified
- Two students were given the same ID 04(Left: correct and Right: Incorrect)

Fig 12: Face Identification with CNN

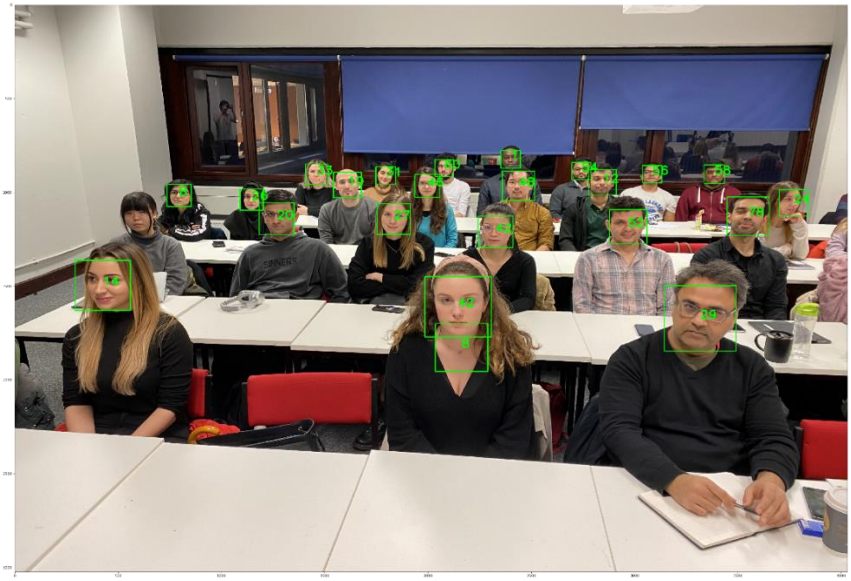
- Three students were wrongly classified.
- At a threshold of 0.15, the model failed to recognise unknown students and unknown students were given ID's
- Viola Jones Algorithm mistakenly identified faces on 'non-face' objects (as highlighted).

Fig 13: Face Identification with Hog-SVM



- Only one student was wrongly classified (ID 27 as ID 31)
- Unknown student was correctly identified, however, a known student was given unknown label
- Viola Jones Algorithm mistakenly identified faces on 'non-face' objects(as highlighted)
- Viola-Jones algorithm failed to identify a face(as highlighted)

Fig 14: Face Identification with CNN



- While only two students were wrongly classified, however as above, CNN failed to recognise unknowns.

While with CNN misclassification error was lower, at a threshold of 0.15, while SVM did a better job than CNN, at predicting 'unknown' ID's. However, SVM gave many known ID's 'unknown' tag.

❖ Image Recognition Function:

PREDICTION PIPELINE: We have constructed an image recognition Function, which when tested on a group image (or an individual image) will detect faces via Vio-Jones algorithm followed by extracting

features-global or local features, followed by prediction of the image ID using any of the four classific ation methods. If feature type is 'none'; it will predict student ID using CNN.

The function accepts the following arguments: Model: 'LR', 'SVM', 'RF', 'KNN', 'CNN'. - Features: 'hog', 'globalfeatures', 'none'. Further an additional argument Creative Mode:'0' or '1' allows us to detect faces and make cartoons of every face, as shown below.

viii) Way Forward:

- Hsu and Jain (2002) proposed a novel lighting compensation technique and a nonlinear color transformation, to detect skin regions over the entire image and that generated face candidates based on the spatial arrangement of these skin patches. The algorithm constructed eye, mouth, and boundary maps for verifying each face candidate. Going forward, this could be an interesting application to the field of above-mentioned face detection problem, which would lead to more success in detecting unknowns in heterogenous environments (11).
- While in our study local feature extraction outperformed global features; Shan and Gao (2010) proposed a fine technique for integrating global and local facial features in parallel manner using Discrete Fourier Transform (DFT) and Principal Component Analysis (PCA) (9). This would be a vital tool, since both set of features offer different information about the image and their amalgamation is essential for the holistic extraction of image features.

References

1) Wang, Y., 2014. An Analysis of the Viola-Jones Face Detection Algorithm. *Image Processing On Line*, 4, pp.128-148.
2) Kamarainen, J., Kyrki, V. and Kälviäinen, H., 2007. Local and global Gabor features for object recognition. *Pattern Recognition and Image Analysis*, 17(1), pp.93-105.
3) Shapiro, L. and Stockman, G., 2001. *Computer Vision*. Upper Saddle River, NJ: Prentice Hall.
4) Xu, D. and Li, H., 2008. Geometric moment invariants. *Pattern Recognition*, 41(1), pp.240-249.
5) Haralick, R., 1973. Glossary and index to remotely sensed image pattern recognition concepts. *Pattern Recognition*, 5(4), pp.391-403.
6) Sperandei, S. (2014). Understanding logistic regression analysis. Biochemia medica: Biochemia medica, 24(1), 12-18.
7) Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010 (pp. 177-186). Physica-Verlag HD.E
8) Latkowski, T. and Osowski, S., 2017. Gene selection in autism – Comparative study. Neurocomputing, 250, pp.37-44.
9) SU, Y., SHAN, S., CHEN, X. and GAO, W., 2010. Integration of Global and Local Feature for Face Recognition. *Journal of Software*, 21(8), pp.1849-1862.
10) Kamarainen, J., Kyrki, V. and Kälviäinen, H., 2007. Local and global Gabor features for object recognition. *Pattern Recognition and Image A*nalysis, 17(1), pp.93-105.
11) Rein-Lien Hsu, Abdel-Mottaleb, M. and Jain, A., 2002. Face detection in color images. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(5), pp.696-706
12) Ilango, G., 2020. Image Classification Using Python And Scikit-Learn. [online] Gogul Ilango. Available at: https://gogul.dev/software/image-classification-python
13) DataCamp Community. 2020. Convolutional Neural Networks In Python. [online] Available at: https://www.datacamp.com/community/tutorials/convolutional-neural-networks-python
14) Consulting, A., 2020. Histogram Of Oriented Gradients | Learn Opencv. [online] Learnopencv.com. Available at: https://www.learnopencv.com/histogram-of-oriented-gradients/
15) Brownlee, J., 2020. How To Configure Image Data Augmentation In Keras. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/
16) GeeksforGeeks. 2020. Cartooning An Image Using Opencv - Python - Geeksforgeeks. [online] Available at: <https://www.geeksforgeeks.org/cartooning-an-image-using-opencv-python/>